

Algorithms Of Optimizing Computer Performance

INTRODUCTION:

Although today's computer systems are robust compared to a few years ago (1), the tasks assigned to today's computers are much more complex, and therefore, it is of the utmost importance that the computer system, on the whole, is able to operate at optimal or near optimal performance. One example would be the high CPU speed needed for graphics display in today's online games (2). This paper will address the three main areas of optimization in modern computers: program optimization, algorithmic optimization, and cache optimization

PROGRAM OPTIMIZATION

Optimization with computer programs does not usually refer to maximum performance, but rather, to any significant increase in performance. This is true due to the computer programs will be used for different purposes and by different audiences, such as, home, office, factory, health care, etc.

A few example are:

In manufacturing, robotic systems can be controlled by just a few computer technicians.

In wholesaling, an accurate count of inventory items is in real-time.

In aviation, computer systems coordinate the arrivals and departures. (3)

In any of these cases, an increase in the speed of computer performance will likely require more memory, and conversely, an increase in the memory available would require more load on the CPU. At the level of the code itself, there are hundreds of techniques that could be employed to use either less CPU or less system memory. A short list of these is: loop unrolling, loop jamming, loop inversion, strength reduction, and loop invariant computations (4).

Design Level:

For many programs, the best place to begin optimization is at the design level to make the best use of CPU, memory, and IO operations. Clear goals need to be documented and followed or else large problems could bubble up later in the process. Even a small oversight at this level could cause a devastating overload by the time of production, which cannot easily be corrected. At this level, the choice of operating platform and programming language is critical (5).

There are several more levels, but describing each one would be beyond the scope of this paper. for further research, some of those levels are: Source Code level, Build level, Compile level,

and Assembly level.

At which level should the developers optimize then? At every level? Optimization can add code which could lead to lack of clarity, and therefore, difficulty maintaining the code in the future. At the very beginning of development, performance goals should be specified, then only if those goals are violated, should optimization take place.

Algorithm Efficiency

Algorithms are specifically-created procedures or steps that will instruct a computer to solve a particular problem or complete a certain task. In the simplest example, every time you open your web browser, your computer executes a well-defined algorithm to accomplish that one task in precisely the same way every single time (6). Other examples could be sorting numbers, formatting text on a page, or diagnosing automobiles.

Algorithms can be very complex such as an algorithm designed to predict the weather. There are also what is known as 'genetic algorithms', which may sound like it pertains only to biological science, but the actual meaning in this context is fascinating, in that, the genetics of Darwinian biological evolution has a place in computer algorithms (7). A genetic algorithm in computers can, over time, actually evolve better and better solutions to the problem to which it was initially tasked.

Many algorithms meant to solve complex problems are not designed to "think" like a human brain, but rather, mimic how solutions are found in the biological world. There is the Ant Colony Optimization algorithm, (ACO), which attempts to find the best "path" through graphs in the same way that ants find their path to and from their colony (8). There is also the Particle Swarm Optimization, (PSO), algorithm which, to keep it simple, takes a set of possible solutions and arranges them over and over again until the best total solution is found. The method of PSO has similarities with Fish School Optimization, and Bee Swarm Optimization.

Every program that users implement on their computers will have many algorithms in order to accomplish the tasks of that particular software. Many anti-virus programs have advanced algorithms such as the ones mentioned above. Just a few examples would be the Avast, McAfee, Mackeeper, and AVG anti-virus programs.

Cache Optimization Issues

Caching is high-speed computer memory that is situated between the main processor, (CPU), and the main memory. The purpose of the cache is to hold on to frequently utilized data for its next reuse, thus, reducing possible bottlenecks. This keeps the load on the main memory lighter and speeds up the data transfer since cache memory is typically built to be faster than the main memory (10).

There are many aspects to consider in cache optimization, such as, compulsory miss, conflict miss, capacity miss, miss rate, hit rate, latency, and efficiency. This paper will briefly describe the first three of those aspects since the scope is thusly limited. The CPU operates at a higher speed than the cache, and the cache operates at a higher speed than the main memory. Because of those speed difference, there are times when the data does not successfully get stored in the cache. These are called "misses". There are three basic misses: compulsory misses, capacity misses, and conflict misses. Compulsory misses occur as the very first access to the cache, since it is empty, there must be a miss. Capacity misses happen when the cache is not large enough to contain all the blocks of data. These blocks of data will be discarded and retrieved later. Conflict misses are when two or more blocks are mapped to the same area of the cache. This, of course, represents a development error in block placement strategy and is one of the main issues regarding cache optimization.

Conclusion:

As you might have guessed by now, computer performance optimization could be considered a science of its own. Certainly, there are Phd level engineers overseeing the development of operating systems, in fact, there are teams that are coordinated for each project. Still, issues arise, that's just the way of computers.

1. Modern computers vs Old computers

<https://sites.google.com/site/wwwcomputinghistory2013com/modern-computers-vs-old-computers>

2. How fast a processor do you need?

<http://www.alphr.com/features/382873/how-fast-a-processor-do-you-need>

3. TECHNOLOGY IS TRANSFORMING THE U.S. ECONOMY

<https://www.dol.gov/oasam/programs/history/herman/reports/futurework/report/chapter6/main.htm>

4. Optimization of Computer Programs in C

http://icps.u-strasbg.fr/~bastoul/local_copies/lee.html

5. Program optimization

https://en.wikipedia.org/wiki/Program_optimization

6. What is a Computer Algorithm? - Design, Examples & Optimization

<http://study.com/academy/lesson/what-is-a-computer-algorithm-design-examples-optimization.html>

7. Genetic Algorithms: Principles of Natural Selection Applied to Computation

<http://www.astro.cornell.edu/~cordes/A6523/stephanie.forrest.pdf>

8. Ant colony optimization algorithms

https://en.wikipedia.org/wiki/Ant_colony_optimization_algorithms

9. Particle swarm optimization

https://en.wikipedia.org/wiki/Particle_swarm_optimization

10. An Overview of Cache Optimization Techniques and Cache Aware Numerical Algorithms

<http://www.cc.gatech.edu/~bader/COURSES/UNM/ece637-Fall2003/papers/KW03.pdf>

11. An Overview of Hardware Based Cache Optimization Techniques

http://www.academia.edu/19957663/An_Overview_of_Hardware_Based_Cache_Optimization_Techniques